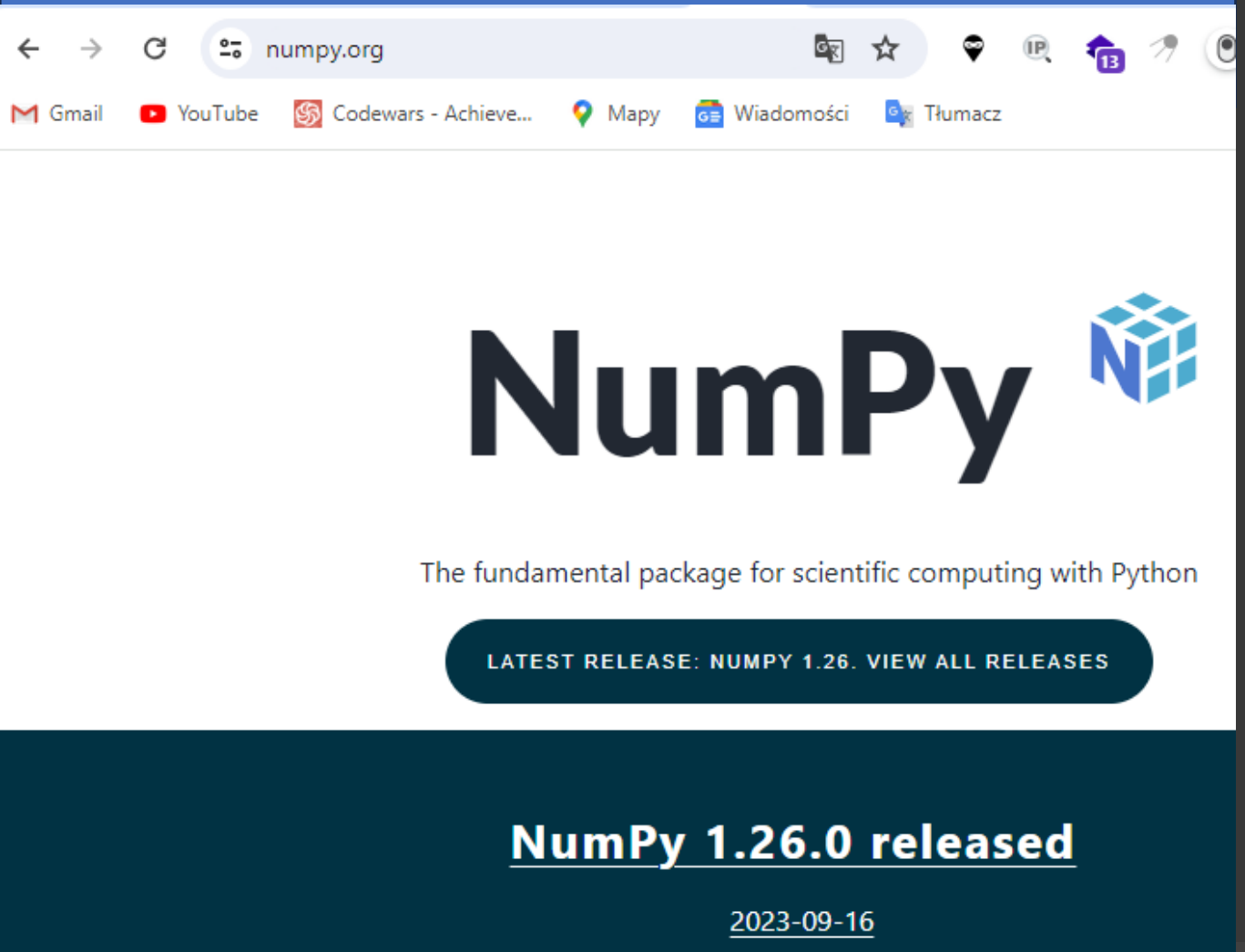
SPIS TREŚCI



```
import numpy as np
np.__version__
```

```
'1.23.5'
```

ROZDZIAŁ I

Operacje na tablicach

I. OPERACJE NA TABLICACH 1 Array Jednowymiarowe

```
x = np.array([1, 3])
x
```

```
array([1, 3])
```

```
print(x)
```

```
[1 3]
```

```
type(x)
```

```
numpy.ndarray
```

```
x.ndim
```

```
1
```

```
x.shape
```

```
(2,)
```

```
x.size
```

```
2
```

```
x.dtype
```

```
dtype('int64')
```

```
np.array([1.3, 2.3, 1.4])
```

```
array([1.3, 2.3, 1.4])
```

```
x = np.array([1.3, 2.3, 1.4])
print(x)
```

```
[1.3 2.3 1.4]
```

```
x.dtype
```

```
dtype('float64')
```

2 Array Dwuwymiarowe

```
[ ]  np.array([[1, 2], [-3, 1]])
```

```
array([[ 1,  2],
       [-3,  1]])
```

```
[ ]  x = np. array([[1, 2], [-3, 1]])
     x
```

```
array([[ 1,  2],
       [-3,  1]])
```

```
[ ]  x.ndim
```

```
2
```

```
[ ]  x.shape
```

```
(2, 2)
```

```
[ ]  x = np.array([[1, 2, 3], [4, 2, 1]])
     print(x)
```

```
[[1 2 3]
 [4 2 1]]
```

+ Kod + Tekst Wszystkie zmiany

```
[ ]  x.shape
```

```
(2, 3)
```

```
x = np.array(
    [[[4, 3, 1],
      [3, 1, 2]],

     [[4, 1, 3],
      [4, 2, 1]]]
)
x
```

```
array([[[4, 3, 1],
        [3, 1, 2]],

       [[4, 1, 3],
        [4, 2, 1]]])
```

```
[ ]  x.ndim
```

```
3
```

```
[ ]  x = np.array(
         [[[4, 3, 1],
           [3, 1, 2]],

          [[4, 1, 3],
           [4, 2, 1]],

          [[3, 3, 1],
           [4, 3, 2]]]
     )
     x
```

```
array([[[4, 3, 1],
        [3, 1, 2]],

       [[4, 1, 3],
        [4, 2, 1]],

       [[3, 3, 1],
        [4, 3, 2]]])
```

```
[ ]  x.ndim
```

```
3
```

```
[ ]  x.shape
```

```
(3, 2, 3)
```

II. TYPY DANYCH

```
[ ] A = np.array([1, 2, 3])
    A.dtype
```

```
dtype('int64')
```

```
[ ] A = np.array([1.0, 2.3, 3.3])
    A.dtype
```

Pokaż ukryte dane wyjściowe

```
[ ] A = np.array([1, 2, 3], dtype='float')
    A.dtype
```

```
dtype('float64')
```

```
[ ] A
```

```
array([1., 2., 3.])
```

```
[ ] A = np.array([1, 2, 3], dtype='complex')
    A.dtype
```

```
dtype('complex128')
```

```
A = np.array([1.0, 2.3, 3.3], dtype='int')
A.dtype
```

```
dtype('int64')
```

```
[ ] A
```

```
array([1, 2, 3])
```

```
[ ] A = np.array([True, False])
    A.dtype
```

```
dtype('bool')
```

```
[ ] A = np.array([24, 120, 230], dtype=np.int8)
    A.dtype
```

```
dtype('int8')
```

```
[ ] A = np.array([24, 120, 230], dtype=np.uint8)
    A.dtype
```

```
dtype('uint8')
```

ROZDZIAŁ II

Typy danych

ROZDZIAŁ III

Tworzenie tablic NumPy ndarray

III TWORZENIE TABLIC NumPy ndarray

```
[ ] np.zeros(shape=(4, 10))
```

```
array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

```
#@title
np.zeros(shape=(4, 10), dtype="int")
```

```
array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
```

```
[ ] np.ones(shape=(5, 5))
```

```
array([[1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.]])
```

```
[ ] np.ones(shape=(5, 5), dtype='int')
```

```
array([[1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1]])
```

```
[ ] np.full(shape=(3, 3), fill_value=4, dtype='int')
```

```
array([[4, 4, 4],
       [4, 4, 4],
       [4, 4, 4]])
```

```
[ ] np.arange(11)
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

```
[ ] np.arange(start=6, stop=11)
```

```
array([ 6,  7,  8,  9, 10])
```

```
[ ] np.arange(start=10, stop=100, step=10)
```

```
array([10, 20, 30, 40, 50, 60, 70, 80, 90])
```

```
[ ] np.arange(start=100, stop=10, step=-10)
```

```
array([100,  90,  80,  70,  60,  50,  40,  30,  20])
```

```
[ ] np.arange(start=0, stop=1, step=0.05)
```

```
array([0.  , 0.05, 0.1 , 0.15, 0.2 , 0.25, 0.3 , 0.35, 0.4 , 0.45, 0.5 ,
       0.55, 0.6 , 0.65, 0.7 , 0.75, 0.8 , 0.85, 0.9 , 0.95])
```

```
[ ] np.linspace(start=0, stop=1, num=10)
```

```
array([0.        , 0.11111111, 0.22222222, 0.33333333, 0.44444444,
       0.55555556, 0.66666667, 0.77777778, 0.88888889, 1.        ])
```

```
[ ] np.linspace(start=0, stop=1, num=11)
```

```
array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ])
```

```
A = np.arange(15)
A
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14])
```

```
A.reshape((3, 5))
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])
```

```
A.reshape((3, -1))
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])
```

```
#@title
A.reshape((-1, 3))
```

```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11],
       [12, 13, 14]])
```

+ Kod + Tekst Wszystkie zmiany zostały zapisane

IV. PODSTAWOWE OPERACJE NA TABLICACH

```
#@title
A = np.array([3, 1, 4, 2])
B = np.array([3, -1, 3, 3])
print(A)
print(B)
```

```
[3 1 4 2]
[ 3 -1  3  3]
```

```
[ ] A + B
```

```
array([6, 0, 7, 5])
```

```
[ ] A - B
```

```
array([ 0,  2,  1, -1])
```

```
[ ] A * B
```

```
array([ 9, -1, 12,  6])
```

```
[ ] A / B
```

```
array([ 1.        , -1.        ,  1.33333333,  0.66666667])
```

```
[ ] np.add(A, B)
```

```
array([6, 0, 7, 5])
```

```
[ ] np.subtract(A, B)
```

```
array([ 0,  2,  1, -1])
```

```
[ ] np.divide(A, B)
```

```
array([ 1.        , -1.        ,  1.33333333,  0.66666667])
```

```
[ ] A + 3
```

```
array([6, 4, 7, 5])
```

```
[ ] 2 * A
```

```
array([6, 2, 8, 4])
```

```
[ ] A + 3*B
```

```
array([12, -2, 13, 11])
```

Aby edytować zawartość komórki, kliknij ją dwukrotnie (lub naciśnij kla

```
X = np.array([[1, 3],[-2, 0]])
Y = np.array([[6, 0], [-1, 2]])
print(X, '\n')
print(Y)
```

```
[[ 1  3]
 [-2  0]]

[[ 6  0]
 [-1  2]]
```

```
[ ] X * Y
```

```
array([[6, 0],
       [2, 0]])
```

ROZDZIAŁ IV

Podstawowe operacje na tablicach

```
[ ] np.dot(X, Y)
```

```
array([[  3,   6],
       [-12,   0]])
```

```
[ ] X.dot(Y)
```

```
array([[  3,   6],
       [-12,   0]])
```

```
[ ] Y.dot(X)
```

```
array([[ 6, 18],
       [-5, -3]])
```

```
[ ] X @ Y
```

```
array([[  3,   6],
       [-12,   0]])
```

+ Kod + Tekst Wszystkie zmiany zostały zapisane

V. GENEROWANIE LICZB PESUDOLOSOWYCH

Aby edytować zawartość komórki, kliknij ją dwukrotnie (lub naciśnij klawisz Enter)

```
np.random.seed(0)
```

```
[ ] np.random.randn()
```

```
1.764052345967664
```

```
[ ] np.random.randn(10)
```

```
array([ 0.40015721,  0.97873798,  2.2408932 ,  1.86755799, -0.97727788,
        0.95008842, -0.15135721, -0.10321885,  0.4105985 ,  0.14404357])
```

```
[ ] np.random.rand(10, 4)
```

```
array([[0.56804456, 0.92559664, 0.07103606, 0.0871293 ],
       [0.0202184 , 0.83261985, 0.77815675, 0.87001215],
       [0.97861834, 0.79915856, 0.46147936, 0.78052918],
       [0.11827443, 0.63992102, 0.14335329, 0.94466892],
       [0.52184832, 0.41466194, 0.26455561, 0.77423369],
       [0.45615033, 0.56843395, 0.0187898 , 0.6176355 ],
       [0.61209572, 0.616934  , 0.94374808, 0.6818203 ],
       [0.3595079 , 0.43703195, 0.6976312 , 0.06022547],
       [0.66676672, 0.67063787, 0.21038256, 0.1289263 ],
       [0.31542835, 0.36371077, 0.57019677, 0.43860151]])
```

```
[ ] np.random.rand()
```

```
0.9883738380592262
```

```
[ ] np.random.rand(10)
```

```
array([0.10204481, 0.20887676, 0.16130952, 0.65310833, 0.2532916 ,
       0.46631077, 0.24442559, 0.15896958, 0.11037514, 0.65632959])
```

```
np.random.rand(10, 2)
```

```
array([[0.13818295, 0.19658236],
       [0.36872517, 0.82099323],
       [0.09710128, 0.83794491],
       [0.09609841, 0.97645947],
       [0.4686512 , 0.97676109],
       [0.60484552, 0.73926358],
       [0.03918779, 0.28280696],
       [0.12019656, 0.2961402 ],
       [0.11872772, 0.31798318],
       [0.41426299, 0.0641475 ]])
```

```
[ ] np.random.randint(10)
```

```
2
```

```
[ ] np.random.randint(low=10, high=101)
```

```
56
```

```
[ ] np.random.randint(low=10, high=102, size=8)
```

```
array([30, 91, 60, 37, 24, 51, 68, 75])
```

```
[ ] np.random.choice([5, 10, 14, 3])
```

```
5
```

ROZDZIAŁ V

Generowanie liczb pseudolosowych

```
[ ] np.random.choice(['python', 'java', 'sql', 'C#'])
```

```
'sql'
```

```
[ ] data = np.arange(10)
    data
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
[ ] np.random.shuffle(data)
```

```
[ ] data
```

```
array([1, 5, 9, 4, 0, 7, 2, 3, 8, 6])
```

VI. FUNKCJE W BIBLOTECE NUMPY

```
[ ] np.exp(1)
```

```
2.718281828459045
```

```
[ ] np.sqrt(9)
```

```
3.0
```

```
[ ] np.all([2, 3, 1])
```

```
True
```

```
[ ] np.any([1,3, 4])
```

```
True
```

```
[ ] np.any([0, 0, 0])
```

```
False
```

```
[ ] A = np.random.rand(5)
    A
```

```
array([0.26473016, 0.39782075, 0.55282148, 0.16494046, 0.36980809])
```

```
[ ] np.argmax(A)
```

```
2
```

```
[ ] A[np.argmax(A)]
```

```
0.5528214798875715
```

```
[ ] np.argmin(A)
```

```
3
```

```
[ ] np.argsort(A)
```

```
array([3, 0, 4, 1, 2])
```

```
[ ] np.max(A)
```

```
0.5528214798875715
```

```
[ ] np.min(A)
```

```
0.16494046024188413
```

```
[ ] np.mean(A)
```

```
0.35002418998397483
```

```
[ ] np.median(A)
```

```
0.36980809274834003
```

```
[ ] np.std(A)
```

```
0.13063974345454746
```

ROZDZIAŁ VI

Funkcje w bibliotece NumPy

VII. INDEKSOWANIE I WYCINANIE TABLIC

```
[ ] A = np.arange(20)
    A
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

```
[ ] A[2]
```

```
2
```

```
[ ] A[2:]
```

```
array([ 2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19])
```

```
[ ] A[:2]
```

```
array([0, 1])
```

```
[ ] A[[0, 2]]
```

```
array([0, 2])
```

```
[ ] A[-1]
```

```
19
```

```
[ ] A[11:15]
```

```
array([11, 12, 13, 14])
```

```
[ ] A = A.reshape(4, 5)
    A
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19]])
```

```
[ ] A[0]
```

```
array([0, 1, 2, 3, 4])
```

```
[ ] A[1]
```

```
array([5, 6, 7, 8, 9])
```

```
[ ] A[:,0]
```

```
array([ 0,  5, 10, 15])
```

```
[ ] A[:, -1]
```

```
array([ 4,  9, 14, 19])
```

```
[ ] A[:, -2]
```

```
array([ 3,  8, 13, 18])
```

```
[ ] A[1, 1]
```

```
6
```

```
[ ] A[1, 3]
```

```
8
```

```
[ ] A[1:3, 1:4]
```

```
array([[ 6,  7,  8],
       [11, 12, 13]])
```

```
[ ] A[1, 2] = 14
    A
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6, 14,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19]])
```

ROZDZIAŁ VII

Indeksowanie i wycinanie tablic

ROZDZIAŁ VII

Iteracja po tablicach
Zmiana wielkości
Maski logiczne

A. ITERACJA PO TABLICACH

```
[ ] A[1, 2] = 14
    A
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6, 14,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19]])
```

```
[ ] for row in A:
        print(row)
```

```
[0 1 2 3 4]
[ 5  6 14  8  9]
[10 11 12 13 14]
[15 16 17 18 19]
```

```
[ ] for row in A:
        print(row[0])
```

```
0
5
10
15
```

```
[ ] for row in A:
        print(row[:3])
```

```
[0 1 2]
[ 5  6 14]
[10 11 12]
[15 16 17]
```

```
[ ] for item in A.flat:
        print(item)
```

```
0
1
2
```

B. ZMIANA ROZMIARU TABLIC

```
[ ] A
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6, 14,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19]])
```

```
[ ] A.shape
```

```
(4, 5)
```

```
[ ] A.reshape(5, 4)
```

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6, 14],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15],
       [16, 17, 18, 19]])
```

```
[ ] A.ravel()
```

```
array([ 0,  1,  2,  3,  4,  5,  6, 14,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

```
[ ] A.T
```

```
array([[ 0,  5, 10, 15],
       [ 1,  6, 11, 16],
       [ 2, 14, 12, 17],
       [ 3,  8, 13, 18],
       [ 4,  9, 14, 19]])
```

C. MASKI LOGICZNE

```
[ ] A = np.arange(start=-10, stop=10, step=0.5)
    A
```

```
array([-10. ,  -9.5,  -9. ,  -8.5,  -8. ,  -7.5,  -7. ,  -6.5,  -6. ,
       -5.5,  -5. ,  -4.5,  -4. ,  -3.5,  -3. ,  -2.5,  -2. ,  -1.5,
       -1. ,  -0.5,   0. ,   0.5,   1. ,   1.5,   2. ,   2.5,   3. ,
        3.5,   4. ,   4.5,   5. ,   5.5,   6. ,   6.5,   7. ,   7.5,
        8. ,   8.5,   9. ,   9.5])
```

```
A = A.reshape(10, -1)
A
```

```
array([[-10. ,  -9.5,  -9. ,  -8.5],
       [ -8. ,  -7.5,  -7. ,  -6.5],
       [ -6. ,  -5.5,  -5. ,  -4.5],
       [ -4. ,  -3.5,  -3. ,  -2.5],
       [ -2. ,  -1.5,  -1. ,  -0.5],
       [  0. ,   0.5,   1. ,   1.5],
       [  2. ,   2.5,   3. ,   3.5],
       [  4. ,   4.5,   5. ,   5.5],
       [  6. ,   6.5,   7. ,   7.5],
       [  8. ,   8.5,   9. ,   9.5]])
```

```
[ ] A > 0
```

```
array([[False, False, False, False],
       [False, False, False, False],
       [False, False, False, False],
       [False, False, False, False],
       [False, False, False, False],
       [False,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True]])
```

```
[ ] A[A > 0]
```

```
array([0.5, 1. , 1.5, 2. , 2.5, 3. , 3.5, 4. , 4.5, 5. , 5.5, 6. , 6.5,
       7. , 7.5, 8. , 8.5, 9. , 9.5])
```

```
np.bitwise_and(A > -5, A < 5)
```

```
array([False, False, False, False, False, False, False, False, False,
       False, False,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True, False, False, False, False, False, False,
       False, False, False, False])
```

```
[ ] A[np.bitwise_and(A > -5, A < 5)]
```

```
array([-4.5, -4. , -3.5, -3. , -2.5, -2. , -1.5, -1. , -0.5,  0. ,  0.5,
        1. ,  1.5,  2. ,  2.5,  3. ,  3.5,  4. ,  4.5])
```

```
[ ] np.bitwise_or(A < -5, A > 5)
```

```
array([ True,  True,  True,  True,  True,  True,  True,  True,  True,
        True, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False,  True,  True,  True,  True,  True,
        True,  True,  True,  True])
```

```
[ ] A[np.bitwise_or(A < -5, A > 5)]
```

```
array([-10. ,  -9.5,  -9. ,  -8.5,  -8. ,  -7.5,  -7. ,  -6.5,  -6. ,
       -5.5,   5.5,   6. ,   6.5,   7. ,   7.5,   8. ,   8.5,   9. ,
        9.5])
```